CS 134: Networks	Lecture 12, Monday, 3/20
Prof. Yaron Singer	Spring 2017

1 Overview

In the previous lectures we discussed models of influence and contagion in networks. We proved various structural properties about the parameters of the models and network structure that ensured that an influence process infects the entire population. In the next two lectures we will be concerned with the algorithmic question of how to select the seed set of initial adopters so that influence in the network is maximized.

Question: *How do we select influencers in a network?*

The main result we will discuss today is due to Kempe, Kleinberg and Tardos [2].

2 The Model

The local influence model. Given a finite edge-weighted directed graph $G = (V, E, \mathbf{p})$ where $\mathbf{p} \in [0,1]^{|E|}$, let $p_{u,v}$ denote the weight encoded on the edge $(v,u) \in E$, and for a given node $u \in V$, let $N(u)$ be the set of neighbors of u, i.e. $N_u = \{v \ : \ (v,u) \in E\}$. To consider the global spread of influence in the network according to this model given a seed of nodes S who are initially influenced, let $I_t(S)$ be the set of nodes influenced at time t. In this case the model is recursively defined as follows. At time step $t = 0$, $I_t(S) = S$. For any step $t > 0$, a given node $u \in V$ is influenced independently with probability $p_{v,u}$ by every one of its neighbors who were influenced at time step $t-1$, i.e. every neighbor $v \in N(u) \cap I_{t-1}(S)$.

When we consider this model spreading in the network, then given some subset of nodes that is initially infected at time step 0, at each step t every node that was infected in $t-1$ has a single chance to infect every one of its neighbors with the probability encoded on the edges. After $n = |V|$ this process terminates[1], and results in some subset of nodes that are infected.

The initial set that we select at time step 0 can have a significant effect on the size of the resulting cascade: the expected number of nodes that will be influenced as a result. The natural question is which set we should select to maximize the cascade.

3 Influence Maximization in the Independent Cascade Model

Influence maximization is the task of selecting k individuals in the network s.t. the expected influence in the network will be maximized after t steps. We can phrase the influence maximization question in the independent cascade model for the case where $t = |V|$, when the process terminates (all the results that we will discuss in this lecture can be easily modified for an arbitrary selection of t).

[1]The process terminates after $|V|$ steps since each node only has one chance to infect its neighbors, and that the edge distance between any two nodes in the graph is at most $|V|$.

3.1 The influence function

A clean way of thinking about influence maximization is as the following optimization problem. Given a subset of nodes $S \subseteq V$, we can define the function:

$f(S) :=$ expected number of nodes influenced after $|V|$ time steps when S is the set of initial adopters

The task of influence maximization then becomes that of maximizing the function under a cardinality constraint. That is, we wish to solve:

$$\max_{S:|S|\leq k} f(S)$$

Instead of thinking about influence maximization, we will instead think of maximizing *submodular functions*. We will later see that the influence function we care about is a special case of a submodular function, and thus the techniques for submodular maximization solve our problem.

4 Submodular Functions

A function is *submodular* if it has a diminishing returns property. More formally:

Definition. *A function $f : 2^N \to \mathbb{R}$ is **submodular** if $\forall S \subseteq T \subseteq N$ and $a \notin T$ we have:*

$$f(S \cup a) \geq f(T \cup a)$$

In words, the definition above says that the marginal contribution of adding an element a to a set S is larger, then the marginal contribution of adding that same element a to a set T which includes S. That is, as S "grows" into a set T, the marginal contribution of adding elements cannot increase.

Definition. *Given a function $f : 2^N \to \mathbb{R}$, the **marginal contribution** of an element $e \in N$ to $S \subseteq N$ is $f_S(e) = f(S \cup e) - f(S)$.*

In the lecture today, we will focus on *monotone* functions.

Definition. *A function $f : 2^N \to \mathbb{R}$, is **monotone** if $S \subseteq T \implies f(S) \leq f(T)$.*

4.1 Examples of submodular functions

- **Additive functions.** $f(S) = \sum_{a \in S} f(a)$ is a called an additive function. It is easy to verity that for this function, for any S and $a \notin S$ we have that $f_S(a) = f(a)$. Thus, the function is submodular since $f(a) = f_S(a) \geq f_T(a) = f(a)$;
- **Coverage functions.** Suppose we are given a set of *circles* $\{T_1, \ldots, T_n\}$ where each circle covers an arbitrary set of points. A *coverage* function is defined as:

$$f(S) = |\cup_{i \in S} T_i|.$$

There are many examples of problems that can be represented as various forms of submodular maximization. For concreteness, we can think of the MAX-COVER problem. In this problem we are given a family of circles $\{T_1, \ldots, T_n\}$ each covering an arbitrary set of points, and a parameter k, and are asked to find k circles which maximizes the number of points covered, i.e. find $\text{argmax}_{S:|S|\leq k} |\cup_{i\in S} T_i|$. here $f(S) = |\cup_{i\in S} T_i|$, and it easy to verify that this function is indeed non-negative, monotone, and submodular.

An algorithm for monotone submodular maximization. We will now see a very simple algorithm which surprisingly has a very good approximation guarantee.

Algorithm 1 Greedy Algorithm

```
Set S = ∅
while |S| ≤ k do
    S ← S ∪ argmax_{a∈N} f_S(a)
end while
return S
```

Analysis of the greedy algorithm. The above algorithm is simple and intuitive: at every step, simply add the element that has the largest marginal contribution to the set of elements selected. We will now show that the algorithm has a good approximation ratio. In particular, we will show:

Theorem 1 ([3]). *For any non-negative monotone submodular function $f : 2^N \to \mathbb{R}_+$, define $\texttt{OPT} = \max_{|T|\leq k} f(T)$. Then, the greedy algorithm returns a solution S s.t. $f(S) \geq (1 - 1/e)\texttt{OPT}$.*

Before we prove the theorem, it will be useful to state the following facts about submodular functions. We will prove these facts in section this week.

- If f, g are monotone submodular functions, and $\alpha, \beta > 0$ then:

  $$h(S) = \alpha f(S) + \beta g(S)$$

  is a monotone submodular function as well;

- A function $f : 2^N \to \mathbb{R}$ is submodular if and only if for every $S \subseteq N$ the marginal contribution function $f_S(T) = f(S \cup T) - f(T)$ is subadditive;

Definition. *A function $f : 2^N \to \mathbb{R}$ is **subadditive** if for any $S, T \subseteq N$ we have that:*

$$f(S \cup T) \leq f(S) + f(T).$$

Notation. At every step i of the while loop in the algorithm, we will use a_i to denote the element that has been selected by the algorithm in that step, and S_i to denote the set of elements the algorithms selected thus far $S_i := \{a_1, \ldots, a_i\}$. Finally, we will use O to denote the set of elements in the optimal solution.

Lemma 2. *At any step $i \in [k]$ we have that: $f(S_{i+1}) - f(S_i) \geq \frac{1}{k}\Big(f(O) - f(S_i)\Big)$*

Proof. Let $O = \{o_1, \ldots, o_\ell\}$, and $o_{\max}$ be the element with the highest marginal contribution in O at stage $i+1$. That is: $o_{\max} = \text{argmax}_{o \in O} f_{S_i}(o)$. At stage $i+1$ the algorithm selects element a_{i+1} and we are guaranteed that its marginal contribution is the highest. In particular, its marginal contribution is also higher than the marginal contribution of the element in O that has the highest marginal contribution: $f_{S_i}(a_{i+1}) \geq f_{S_i}(o_{\max})$. Therefore:

$$
\begin{aligned}
f_{S_i}(O) &\leq \sum_{j=1}^{\ell} f_{S_i}(o_j) && (1)\\
&\leq k f_{S_i}(o_{\max}) && (2)\\
&\leq k f_{S_i}(a_{i+1}) && (3)\\
&= k \cdot \Big(f(S \cup a_{i+1}) - f(S_i) \Big) && (4)\\
&= k \cdot \Big(f(S_{i+1}) - f(S_i) \Big) && (5)
\end{aligned}
$$

Where the first inequality is due to subadditivity of f_{S_i} (which we have from submodularity of f_{S_i}), and the second inequality is due to the fact that the optimal solution has at most K elements.

We therefore have:

$$
k \cdot \Big(f(S_{i+1}) - f(S_i) \Big) \geq f_{S_i}(O) = f(S_i \cup O) - f(S_i) \geq f(O) - f(S_i)
$$

as requried. □

Lemma 3. *At every step $i \in [k]$ we have that: $f(S_i) \geq \Big(1 - (1 - \frac{1}{K})^i\Big) f(O)$.*

Proof. The proof is by induction on i. For $i = 1$ we have that $S_i = a_i$ and we know, using the same argument we used in the proof of the lemma above that $f(a_j) \geq \frac{1}{k} f(O)$. Therefore for $i = 1$:

$$
f(S_i) = f(a_j) \geq \frac{1}{k} f(O) = \Big(1 - (1 - \frac{1}{k})\Big) f(O)
$$

We can now assume the claim holds for $i = \ell$ and we will prove that it holds for $i = \ell + 1$.

$$
\begin{aligned}
f(S_{\ell+1}) &\geq \frac{1}{k}\Big(f(O) - f(S_\ell) \Big) + f(S_\ell) && (6)\\
&= \frac{1}{k} f(O) + \Big(1 - \frac{1}{k}\Big) f(S_\ell) && (7)\\
&\geq \frac{1}{k} f(O) + \Big(1 - \frac{1}{k}\Big)\Big(1 - \Big(1 - \frac{1}{k}\Big)^{\ell}\Big) f(O) && (8)\\
&\geq \frac{1}{k} f(O) + \Big(1 - \frac{1}{k}\Big) - \Big(1 - \frac{1}{k}\Big)^{\ell+1}) f(O) && (9)\\
&\geq \frac{1}{k} f(O) - (\frac{1}{k}) f(O) + \Big(1 - \Big(1 - \frac{1}{k}\Big)^{\ell+1}) f(O) && (10)\\
&= \Big(1 - (1 - \frac{1}{k})^{\ell+1}\Big) f(O) && (11)
\end{aligned}
$$

The first inequality is from Lemma 2. The inequality 8 is due to the inductive hypothesis.

□

Note that for any $k \geq 1$ we have that $(1 - \frac{1}{k})^k \geq \frac{1}{e}$. The theorem now follows. This result is due to Fisher, Nemhauser, and Wolsey [3]. Remarkably, it turns out that the greedy algorithm is optimal: Feige showed that unless P=NP, no polynomial-time algorithm can do better [1].

5 An approximation algorithm for Influence Maximization

Going back to influence maximization, we will now show that maximizing influence in the independent cascade model can be viewed as maximizing a submodular function. To see this let Ω be the set of all graphs with nodes V and edges that realize according to the probability encoded on the edges. For any graph $G \in \Omega$ let $R_G(S)$ be the set of all nodes that are reachable in G from S. For any realization G, the nodes reachable from S in G are the nodes that are influenced in that realization. Therefore, influence in the independent cascade model can be written as:

$$f(S) = \sum_{G \in \Omega} \mathbb{P}[G] \cdot |R_G(S)|$$

The function $|R_G(S)|$ is a cover function, which is monotone submodular. It is easy to see that (positive) weighted sum of submodular functions is itself a submodular function. Therefore, influence in the independent cascade is monotone submodular.

In order to apply the algorithm above we need to be able to evaluate the submodular function. At a first glance, it seems difficult to do since the sum is over exponentially many cover function. This however, can be handled through various approaches of sampling. In the problem set this week we will see one simple method to apply the greedy algorithm with sampling in a way that can give us results that are arbitrarily close to $1 - 1/e$.

References

[1] U. Feige A threshold of ln n for approximating set cover. *Journal of the ACM*, 45, 634 652.

[2] D. Kempe, J. Kleinberg, E. Tardos. Maximizing the Spread of Influence through a Social Network. *ACM SIGKDD*, 2003. http://www-bcf.usc.edu/∼dkempe/publications/spread.pdf

[3] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, 14 (1978), 265294.